

ACE ATARI  
COMPUTER  
ENTHUSIASTS

3662 Vine Maple Dr. Eugene OR 97405

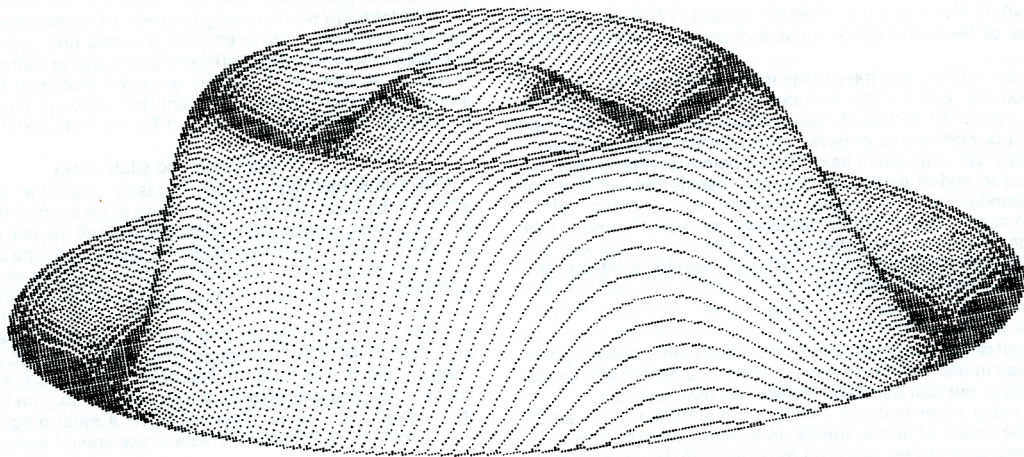
---

JUNE, 1984

EDITORS:

Mike Dunn, Jim Bumpas, Larry Gold

---



**graphics dump from MEGAFONT**



# News and Reviews

by Mike Dunn, Co-Editor

Because of a need for a really professional word-processor, I recently bought WordStar for my CP/M equipped ATR 8000. To make it easier to use, I bought a **WYSE 50** terminal with a built-in WordStar ROM. This is the most fantastic piece of equipment I have ever used. The screen display is unbelievably clear and easy to read; it displays in either 80 or 136 columns. The keyboard is perfect, and it has 61 labeled function keys for use with WordStar, as well as 13 unlabeled ones. All the function keys can be re-programmed to work as you wish, and a 25th status line can put labels for some of them on the screen. This terminal makes WordStar a breeze to use, and I recommend it to anyone who needs a professional level very intelligent terminal for use with their ATR-8000. The terminal retails for about \$699 plus \$100 for the WordStar ROM. However, I think I bought all this equipment a little too soon, because I just received the new **Letter Perfect**.

## Letter Perfect Version 6

(L.J.K., 7852 Big Bend Blvd., St. Louis, MO 63119, \$99)

This is a major update of what I have always considered the most powerful of Word Processors for the Atari. The first major change is the documentation is first-rate — the one major fault with L.J.K. products in the past. The next thing you will notice is the documentation begins with instructions on how to back up your disk! L.J.K. has never copy-protected programs, saving lots of money which has been put into a fairly continuous update policy. They have been around a long time and seem to be making money, so I guess they feel it's not a problem for them. Your LP comes unconfigured — once the disk is configured, it will boot in fully configured, but can easily be changed at any time. You boot in your new copy, and have a choice of 40 column display, or 80 columns with either the BIT-3 or Austin-Franklin boards. (I understand Austin-Franklin is no longer in business). You then have several menus to choose the defaults you want, such as the drives, **single or double density** and the the printer menu. You have a built in choice of the Centronics, Prowriter, Epson, Qume, NEC, and Okidata printers, as well as the ability to make your own custom printer drivers. There is also an option to make dot-space character tables for **Proportional Print**!! It's easy to do and clearly explained. After all your options have been chosen, they will boot in that way each time until you change it.

The program is still menu driven as is the old LP, but you can use the first letter of the Menu option instead of using the cursor, if you wish.

When you are editing, you have many more commands to use. New cursor movement commands include the ability to advance a paragraph, go word left and word right and jump to a marker. You can delete the left or right words or have many different delete options — all proceed by K for "Kill" such as delete all before or after cursor, up to marker, and to end of paragraph. You can change the width of the screen, use conditional page breaks, imbed printer commands, etc.

There is an easy to use, powerful database merge ability, and, if this is not enough, a built-in, integrated spelling checker!!! This spelling checker is not only very fast and easy to use, it is integrated with the editor, so your words are flagged in the word processor, and you are given a list of possible words to use instead of your mis-spelled word. Just pick the number from the word menu, or type in the word you want. This is all done in memory, so you do not load in a new program, nor do you load in the text file — it is all there. You cannot add words to the dictionary, but can use **Spell Perfect** for that purpose.

There are many other features I have not mentioned, such as an increase in the speed of all the functions working, and the functions LP already had made it very powerful to begin with. It will do almost anything my new WYSE/WordStar can, except for the wonderful display screen and the function keys, for about \$1000 cheaper!! An exceptionally fine product, except it still uses the non-standard LP file format, so you need another program to use Atari files.

L.J.K. also sent me a new version of their **Data Perfect** version 2.02. I have not had time to really go through it well, but the documentation has improved greatly, from the almost incomprehensible to excellent.

## MegaFont

(Xlent Software, POB 5228, Springfield, VA 22150, \$20)

This easy to use program allows you to dump converted character sets to an Epson, Prowriter or Nec printer. It comes with several fonts you can use, or you can use your own as made by various font editors. You can also list the special character to your printing of listings. In the last few months, we have been using PrintWiz, but this month we will use MegaFont. The titles of the listing will also be done by this program; remember these are reduced in size by the printer. See what you think.

## New Atari Magazines

We have received samples of two new Atari magazines. Both are excellent and are for the intermediate to advanced users. **ROM Magazine** (Box 252, Maple Ridge, B.C. V2X 7G1 Canada \$12 year) has an arcade style machine language game with each issue. The other, **The U.K. Atari Computer Club Newsletter** (POB 3, Rayleigh, Essex, U.K. 5 Pounds Sterling for 4 issues) is an absolute delight!! Beautifully done, it is as nice a magazine as you could want. Put out by the largest U.K. User groups by members, it is a very professional work of love. Speaking of Newsletters, the May 1984 issue of Milatari Newsletter Box 1191, Waukesha, WI 53187-1191 has a 16 page article by Eric Hanson on **3-D Graphics** with an extensive program, including a printer dump for a AlphaCom 42 printer. Eric did this for a High School programming contest and won first place. The article and extensively documented program listing is first rate. I spoke to the editor, and he will send ACE Members a reprint for \$2, or a disk with the program and the article on it for \$5. Very much worth it — the program is long, so send for the disk version, unless you really like to type, then print it out and study the listing — you will learn a lot.

## Getting On-Line: A guide to Accessing Computer Information Services

(Prentice-Hall, Englewood, NJ 07632 \$15)

Another very nice book from the Prentice-Hall people, this 306 page guide has everything you need to know to access databases such as CompuServe, Dialog, etc. It covers hardware and software for the Atari as well as many other computers, interfacing the RS-232 ports, and a comprehensive guide to various databases, including prices, features, ease of use, hints, etc. Nicely done, comprehensive and easy to read, as are most of the books from this company.

## MPP-1150 Parallel Printer Interface

(MicroBits, 225 W. 3rd, Albany, OR 97321 \$99)

In the April issue of ACE, we reviewed a number of printer interfaces to use instead of the difficult to get Atari 850. It comes in a nice blue small box with a 34" I/O cable to connect to your Atari serial port (where disk drives, cassettes, etc. are connected) and a 34" ribbon cable with a standard Centronics style connector on the end. You simply connect the I/O cable to your computer, the ribbon cable to your parallel printer, and your disk drive, etc., to the I/O connector on the interface. No power supply is needed. Optional items include a 4K buffer and a serial printer adaptor. It works fine with no fuss, and is compatible with all Atari Software. There are no switches or anything to set up. A very nice unit from our good friends at MicroBits, to go along with their excellent modem. Incidentally, they are also now shipping a memory expansion unit for the Atari 600XL.

## This issue and Club News

This issue features a really fantastic education game written in ACTION! by Stan Ockers which needs to be seen to be believed. The scrolling and graphics are superb. Because of the use of PM graphics, our printer dumps by PrintWiz do not show it all. The Source code for those of you who have ACTION! is now available on our brand-new ACTION! Disk #2, also featuring the ACTION! programs in the Newsletter since ACTION! Disk #1, as well as a number of Utilities and demos by Clint Parker (the Author of ACTION!), and some other unpublished ACTION! programs. These disks are \$15 each, or \$20 for a double sided disk for both. The next "Best of ACE #9" and the next educational disk will also have this program in "run-time" format for those of you who do not have ACTION! A must program for those of you teaching the Alphabet. These disks should be ready in a month or so.

The next meeting of ACE will feature a demonstration of the fabulous new **MindSet** computer. Designed by a team of former Atari people who helped design the Atari, it features an 80186 chip for high speed, excellent graphics, animation and features similar to the Atari ANTIC chip. It also runs almost all IPM-PC software including Lotus 1-2-3. We will show a demo program of the forthcoming game by Synapse designed for the MindSet called **Viper**. This computer is considerably less costly than the IBM-PC or similar computers, and has many more features.

You might recall we reported a rumor last year about North American Phillips buying Atari. Well, we were apparently only half correct on that one. Larry Gold, our Vice-President, has learned North American Phillips (the large, Dutch-owned electronics company) has bought 49% of Atari from Warner Communications. This is exciting news, as the infusion of talent, money and electronics expertise indicated by this news is sure to work to the benefit of Atari users and the market as a whole.



## BUMPAS REVIEWS

**SEVEN CITIES OF GOLD** (Electronic Arts, \$40) is designed by the same folks (Ozark Softscape) who designed M.U.L.E. This is a real-time graphic adventure game for one player. You begin the game in the Spanish Court where you are given a grant of gold with which to equip your first voyage. You may buy ships, men, food, and trade goods at the Outfitter's store. All play action uses the joystick.

The town (Cadiz?) demonstrates some very advanced 3D graphic effects. As you walk down the street, the buildings in the foreground scroll past you faster than those in the background. In the Pub you may save any game in progress (one to a disk). In your Home you may view all the maps you've drawn, total your gold accumulations and review the percentage of land, rivers, natives and special geographic features you've discovered. When you step aboard the ship, the program places you west of the Canary Is. You now have less than 50 years to open up the New World!

As you sail west across the Atlantic, you will notice your food disappearing. This is your real-time clock. One point of food goes for each passing week. You may encounter storms on the voyage, and up to half your food may spoil at any time. The screen shows an 8x8 map with your player symbol in the center. Around the map the screen displays the number of ships, men, gold, goods and food you're carrying. When at sea, a depth gauge tells you "deep", "medium" or "shallow". If it reads "deep", you're not near land.

At any point in the game, the trigger will bring up a menu. With this you may view a larger map which will show water everywhere except where you have already discovered land. When on land, this menu also permits you to unload or pick up times, to trade, or to make gifts. One option also will "amaze" the natives. Pushing the trigger and pointing the joystick at the same time permits changes in speed.

In the New World you will find native civilizations and gold as well as all the geographical wonders. You may build Forts and you may make converts among the natives who may invite you to build Missions. Natives have 3 things you need: Food, gold, and bearers to carry things. Bearers will carry 10 times as much as Spaniards. Bearers will also lead you to gold mines or capital cities. Occasionally bearers runs off, leaving you with more than you can carry. Your touch is fatal to natives. If you kill too many of them, they will attack you and kill your men. You might be able to make amends for accidents with gifts.

You must return to Spain to record the progress of each voyage. If you don't make it back, someone else in your family will apply to the King for another grant of gold to equip a new expedition. The game may be played at Novice, Journeyman, or Master levels. A Novice will be aimed at land upon setting forth in ships, and all villages and cities will appear on the screen as you approach them. At higher levels, villages and cities are invisible. You must stop and look for the lights of their fires. Also the bearers may try to lead you astray in your search for gold. Storms at sea may sink ships and blow you off course. Shallows may also sink ships of a more experienced player.

As you discover more of the New World a pleased monarch will reward you with promotions until you've attained the highest honor: Viceroy. Once you discover the historical New World, the game also provides a program to generate a number of semi-random alternate New Worlds to discover. This program requires 10 minutes to complete the work. The authors have included a plate tectonics model for the geography and a cultural diffusion model to distribute the population. **Seven Cities of Gold** shows us some very advanced coding and great play value. I'm sure it will be an award-winner and may even surpass M.U.L.E. in popularity. Keep up the good work, Ozark Softscape!

Boxing fans might have a lot of fun with **TITLEBOUT** (Avalon Hill, \$30). For one or two players, you may choose from among 500 or so boxers from all weight classes. This game is one of Avalon Hill's "Sports Illustrated" games. You might be familiar with their board game of the same title. This mostly text game is nearly a direct rendition of that game for the Atari.

You can answer all the "what if" questions you desire. Can Jack Johnson beat Muhammad Ali? (I tried 4 times, losing three and drawing one). You can even mix weight classes. The program also has room for you to create your own (imaginary) boxers, or to add new boxers who appear on the scene in the future. The program tracks nearly 3 dozen characteristics of the boxers in order to execute each round of a fight. The stats for these characteristics have been (subjectively) determined by the designers of the game, but you'll probably find they are pretty accurate. If you disagree, well you can add your own "Jack Johnson" (for instance) with modified characteristics.

Written in BASIC, there are some graphics showing two boxers in the ring with a referee. They punch when the program scores hits, they clinch, they get knocked down. A text window shows wise-cracks by the boxers, along with descriptions of the punches being scored. All boxers do the "Ali shuffle".

Players select the boxers to use, and before each round, you may select from among approximately 30 strategies to use. There is no "arcade" action, or player control during a round. The computer takes over at the bell. With this game, you may create all the "dream matches" you might desire.

— Jim Bumpas

## PACK IT and CHAIN IT

by George Schwenk (c) 1984 by SUPERware, 2028 Kingshouse RD., Silver Spring MD 20904

Description: Atari BASIC utilities. Requirements: disk drive, Atari BASIC, 32K RAM for PACK IT, 16K RAM for CHAIN IT. Cost: \$20 each.

Recently, an acquaintance and I were discussing some problems encountered in converting an Apple program to the Atari. The program in question requires the use of two machine language routines — one to compact saved picture files and the other to chain together the several portions making up the complete program. My friend wondered how to write the routines. Did I have a surprise for him!

Just a couple of days before, Jim Bumpas had asked me to review two new programs by George Schwenk, whose previous works include EASY and X BASIC (See April and May 1984 ACE Newsletters for reviews). Mr. Schwenk's new programs, PACK IT and CHAIN IT, are precisely what my friend needs.

To test PACK IT, I put the disk in my drive and turned on my 800 (with no cartridge inserted). In a few seconds, I was asked to type in the name of the data file I wished to pack, so I inserted the graphics disk we recently received from the Jacksonville Florida ACE and typed "D:STARWARS". The data file was then loaded into the computer and I was told it contained 7,684 bytes. After giving a name for the packed file, it was saved on the disk and I was told the new file contained only 2,058 bytes! I then loaded in DOS to compare the files on the disk. To my surprise, whereas the original file required 62 sectors of storage space, the new file needed only 17.

Not every file will pack so nicely, however; the GATOR file on the same graphics demo disk, which started also with 7,684 bytes, packed only to 5,652 bytes and required 46 sectors to save. Mr. Schwenk warns some data files may even be larger when packed than before, depending on the complexity of the data. Nonetheless, packing can easily increase the storage space on a disk.

To use the packed files, you simply add to your basic program a short machine language routine which is initialized from lines 29300-29310. The routine allows you to choose one of three options. The first is to unpack from RAM after loading the packed file into a buffer. The second option loads a file to RAM, but performs no unpacking. This option can be used to move any data file, such as a character set. The final option unpacks from the disk file as it loads directly into a chosen area of RAM. PACK IT works extremely fast and will satisfy anyone needing this feature.

CHAIN IT does just what the name implies. It allows you to save certain variables which can be passed on to other segments of a long program. It does not use page six, so there should be no conflicts with machine language routines you might have stored there. Instead, it uses locations on page zero beginning at \$D4. All you have to do is ENTER a machine language subroutine into all the segments of your BASIC program, GOSUB to it first before starting a subprogram and again just prior to LOADING the next.

Both of these programs are easy to use. The documentation is short, but the examples should be clear enough even for beginners. I can easily recommend them to anyone needing these routines.

— DeLoy Graham

## FLASHING NUMBERS

by Chris Simeral, 54 E. White Terrace, Nutley, NJ 07110

Have you ever wished you could read as fast as those people you see in those speed-reading commercials? Well, Flashing Numbers may not make you a speed reader, but it will improve your reading speed.

What this program does is to train your eyes to look very quickly at something and remember it. It does this by flashing a six digit number on the TV screen for a fraction of a second and then asking you to type in what it was. This really works!

Not long ago, a professional hockey goalie was having trouble blocking shots. He saw a doctor and the doctor told him his eyes may not be fast enough to watch the puck all the way to make the save. He started him on a training plan using a projector to flash numbers on a screen. After a few weeks of doing this for half an hour a day, he went back to his goalie position and his saves improved by 30%!

When I heard this, I immediately set to work designing a program to do something similar to what that machine in the doctor's training program did. After a few trials and errors, I came up with this program. I have been using it for about a month now and have found it really has made my reading much faster. Let's wait until baseball season and see what it will do for my hitting. The program also keeps track of the number of consecutive right and how many right answers were given for the number of attempts.

As your eyes become quicker, the program may become too easy, to make it harder just change the 60 in line 40 to a lower number or add another 9 to the set of nines in line 20 and add another 1 to the set of ones in line 30.



## TYPE-AHEAD

One thing which will improve the already great Atari screen editor is a "type-ahead". It's a gizmo letting you type something in while the computer is busy, then enters it when the computer wants input. This can be useful during LISTing or LOADING a program. Sound tough?

No problem. All we need to do is modify the screen editor to set a flag when it's ready for input. Then we need a VBI routine to check for key presses, and store them. The VBI will also send the characters when the computer is ready.

When typing in the BASIC loader program, be sure to get the data right and SAVE it. When RUN, the program will generate an AUTORUN.SYS file with the type-ahead on it. Now re-boot the Atari, and the type-ahead will load and run. To test it, enter this line when the READY prompt appears: FOR G=0 TO 1 STEP 0:NEXT G. While this one-liner is running in circles, type in anything you want. Then hit BREAK. What you typed will now come up onto the screen. Note you can use all the screen editing functions with this program.

### NOTES:

The type-ahead uses ALL of Page 6 for the main program, and uses the cassette buffer to hold the key presses. This doesn't mean you can't use the cassette, just don't press any keys while you load or save with the cassette.

Type-ahead will work in the DOS menu. Just don't Binary Load anything into Page 6!

A RESET won't kill the type-ahead, but using the B option from DOS will. Do an A=USR(1536) to recover.

— Greg Menke

## NEWDOS

This article by Greg Menke was inadvertently omitted from the May issue of ACE.

One bad thing about Atari error handling is the error numbers instead of error messages. It is quite easy to put your own error handler into DOS. All you need to do is compile a JSR to your error routine into the error handling part of DOS. Then if DOS gets an error it jumps to your routine. Obviously, any part of DOS can be modified this way. There is a problem, however. As the JSRs are hand compiled into DOS, this type of routine is ONLY compatible with the DOS it was made for. If you modify DOS like this, be sure to get an OK from the authors of Atari DOS before you send it anywhere. (Their address is in INSIDE ATARI DOS, by Bill Wilkinson.) Now, using NEWDOS.

First, type in the program, SAVE it and RUN it. A new AUTORUN.SYS containing NEWDOS will be written to disk. Then reboot the computer. NEWDOS will load in and initialize. Then do an "ENTER"D;. The computer should respond with a "FILENAME ERROR" message and an ERROR 165. If it didn't work you typed it in wrong.

This routine uses low memory, so you need a MEM.SAV file to use DUP. This program, besides modifying DOS, modifies part of the MEM.SAV handler so NEWDOS will re-initialize when you do a RUN CARTRIDGE from DOS. So DO NOT do any disk copying, duplicating or file copying, or MEM.SAV will be invalidated and DOS will crash when you do a RUN CARTRIDGE. If you accidentally invalidate MEM.SAV, reboot the computer when you are finished, don't do a RUN CARTRIDGE!

Neat stuff:

POKE 7533 (\$1D6D),0 - Print error  
      ,1 - Don't print error

POKE 7534 (\$1D6E),0 - Normal messages  
      ,128 - Inverse messages

## KRAZY CLIMBER

In this month's game by Sydney Brown, you must climb to the top of the building without being hit by the falling tools or other objects. You can climb only on the windows and you must also be careful not to fall off the edge of the building.

Joystick 1 controls the climber. He will move in the direction of the joystick. You are allowed 2 mistakes. The third one is FATAL!! You score 1 point per floor climbed and also bonus points when you reach the top of the building. Good luck!!

— Sydney Brown

## ALPHABET BEE

Alphabet Bee is a program to help pre-schoolers learn the alphabet. The object is to pick letters in alphabetical order and carry them to the hives at the right and left limits of travel. If the wrong letter is brought, it will return to another flower. Use a joystick in player #1 position and press the fire button to pick up or release the letters. If you have difficulty picking up the letters, you can hold the button down while you 'buzz' around the flower. The letters can't be released until you reach the limits of travel. You can restart at any time by using the START key. The OPTION key will toggle on and off a listing of the alphabet to copy.

The program uses a number of techniques you might like to investigate. Horizontal scrolling is provided by installing a display list in page six. Action! has no trouble in changing the LMS addresses fast enough to scroll quite smoothly. A single line player missile graphics area is placed at the top eight pages of memory. The ten pages below this are allocated to screen data, each page holding data for two horizontal lines. Memory top is lowered for this 18 pages before a Graphics 1 command is issued. This places the text area 180 bytes below the new memtop. The graphics 1 screen created is never used, but information is put in the text area which the display list has changed to graphics 1 lines.

A new character set is created in the 1K area unused by the players. All letters are made in inverse so they blend with the flowers. Display list interrupts are used to change flower colors part way down the screen. Text area colors are also changed with DLI's. The interrupt routine is placed in page six and has the ability to change two colors for each DLI. I found much interference between the DLI's and player missile graphics. In certain cases the interrupts had to be disabled as when the alphabet is put into the text area or when the program is restarted using the START key.

Timing is largely based on waiting for the vertical blank interrupt (until vcount = 128). This gives the basic speed to movement as well as tempo to the tune played. The notes are made to trail off in amplitude after the count for their duration reaches a certain value. Sound routines were set up so other tunes or sequences of tones could be added. The data for these must be transferred to the array tune1 or tune2, (voices 1&2), before they are activated (sndflg1 = 1). I hope study of the techniques used proves valuable to you.

— Stan Ockers

## SUPERWARE GAMES

**AFTER PEARL** (\$20) is a world war II simulation of the Pacific theater, in which every major ship is depicted. Major ships include cruisers, battleships, and carriers. The simulation begins with the bombing of Pearl Harbor, and may be played as a 2 player simulation or solitaire. Units are shown as red or blue flags of the respective countries, thus hiding task force compositions. Points are scored by the number of currently controlled bases. The map displays the Pacific from Hawaii to China, Alaska to Australia, and is quite detailed. It scrolls smoothly across approximately dozen screens.

There are various levels of play, but even on the advanced level, it is quite easy to win against the computer. However as a 2 player game this program is quite good. All player input is by joystick as you give orders to task forces to move. Combat occurs when task forces come near each other or land bases. There are a couple dozen disturbing misspellings of ship and base names. But this carelessness does not extend to the program's coding. The game is an accurate strategic simulation, and a fun game at the same time. It allows you to play a strategic level game in quite a bit of detail, while still requiring little play time.

At first glance, **DAWN OF CIVILIZATION** (\$11) appears a simple game. However, after repeated plays I find it anything but simple. The game may be played by up to 4 persons. The Atari will play any remaining position. The winner is the one with the most tribes at the end of the game. Players may set the length of the game from 11 to 30 turns. Each turn a player may place a Tribe, Farm, or Legion on the 12 x 12 play area.

Any three Tribe symbols adjacent to an empty space produces a new tribe symbol. Each Tribe and Legion require the food produced by 4 adjacent empty spaces in order to survive each turn. Each turn you have a growth and death cycle. The Farm symbol produces food equal to 2 empty spaces. The Legion symbol allows you to destroy Farms and protect your own areas. A player may not place Tribes or Legions adjacent to another player's Legion.

This is also a good solitaire game, with the computer playing from 1-4 positions. If there is any problem with the game it is in the limited graphics, and small playing area.

— Nick Chrones



## Ruth Ellsworth Con't

```

10 *SAVECHR
20 C:0B764=255
30 T:ENTER FILENAME
40 T: D:filename C:
50 T:
60 A:$FILENAME
70 C:$A=NA
80 C:$B=NB
90 C:$C=NC
100 C:$D=ND
110 C:$E=NE
120 C:$F=NF
130 C:$G=NG
140 C:$H=NH
150 WRITE:$FILENAME,$A,$B,$C,$D,$E,$F,$G,$H
160 CLOSE:$FILENAME
170 E:
180 *RETRCHR
190 T:ENTER FILENAME
200 T:D:filename C:
210 T:
220 A:$FILENAME
230 READ:$FILENAME,$STRING
240 CLOSE:$FILENAME
250 C:NT=0
260 *COUNTER
270 A:=$STRING
280 MS:|,
290 A(NT=0):NA=$LEFT
300 A(NT=1):NB=$LEFT
310 A(NT=2):NC=$LEFT
320 A(NT=3):ND=$LEFT
330 A(NT=4):NE=$LEFT
340 A(NT=5):NF=$LEFT
350 A(NT=6):NG=$LEFT
360 A(NT=7):NH=$RIGHT
370 C:$STRING=$RIGHT
380 C:NT=NT+1
390 J(NT=8):*COUNTER
400 J(NT=8):*BEGIN
410 E:

```

```

1 R:RHYME TYPE
10 R:ACE NEWSLETTER
20 R:JUNE 1984
30 R:Ruth Ellsworth
35 *BEGIN
37 T:M
40 T:TYPE A WORD TO BE RHYMED
50 A:$WORD
55 *TEST
60 MS:A,E,I,O,U
70 C:$FIRST=$MATCH$RIGHT
80 A:=$RIGHT
90 MS:A,E,I,O,U
100 CN:$MORPH=$FIRST
110 C:$MORPH=$MATCH$RIGHT
120 T:TYPE A WORD THAT RHYMES WITH $WORD
130 A:$RHYME
140 MS:$MORPH
150 JM:*RIGHT
160 TN:$RHYME DOES NOT RHYME WITH $WORD. TRY AGAIN.
162 T:
163 T:
165 J:*TEST
170 E:
200 *RIGHT
210 T:YOU DID IT! $RHYME RHYMES WITH $WORD.
215 PA:300
220 J:*BEGIN
230 E:

```

June Meeting  
June 13th, 7:30PM  
South Eugene High  
Cafeteria

## TYPE AIDS

The strings in Krazy Klimber by Sidney Brown contain the following characters (I use the convention "C-" for Control characters, "I-" for inverse, "S-" for Shift):

LINE 7 AS= C-, C-, S-+ S-+ H Esc-Tab 2-down arrows S- = \$ d C-D C-F C-, C-,

LINE 7 BS= :: C-R Esc-C-Delete I-8 I-8

\$ & C-, C-, C-,  
LINE 7 DS= C-, C-X ! I-? I-x ? ! C-X C-,  
LINE 7 GS= C-, C-, @ I-C- @ @ Esc-Delete Esc-C-Insert Esc-C-Insert  
LINE 7 AMS= C-, C-, right-arrow ; Q tEsc-C-Insert Esc-C-Insert B C-, C-,  
LINE 8 HS= @ I-C-, I-S- = I-C-,  
LINE 8 SS= C-C C-G C-F C-H C-P  
LINE 8 S1\$= C-H C-P I- ( @ I-C-,  
LINE 8 S2\$= ! left-arrow !  
LINE 8 PIS= C-P C-J C-L ) C-Z C-L C-H ? ? 3-left-arrows  
LINE 8 ES= C-X \$ B B Esc-C-Insert Esc-C-Insert Esc-C-Insert Esc-C-Insert  
Delete Esc-Delete Esc-Delete Clear  
LINES 305, 348, and 370 the characters are all C-,

— J. B.

The new MINDSET computer  
will be shown at the  
meeting. WOW!!!



# KRAZY KLIMBER by Sydney Brown

```

1 ? "K":High=0
5 Dim P$(1),Z$(380),A$(15),B$(13),D$(9),S$(6),S1$(5),S2$(3),H$(4),P1$(12),X$(12),E$(11),G$(9),AM$(11)
7 A$="♥♥\H)++|Sd|/♥♥":B$=":-[BB)$& ♦♥♥":D$="♥+!23?+!♥":G$="♥♥0004[1]":AM$="♥♥;0[1]8♥♥"
8 H$="000":S$="4V/A ":S1$="400":S2$="!+!":P1$="A") M/?+?+?+?":E$="5BB[1]111<"
10 Vx=Peek(134)+256*Peek(135):Ux=Peek(140)+256*Peek(141)
20 Graphics 2:Poke 710,0:Poke 708,222:
? #6;"-----";? #6;"
  Krazy Klimber"
21 Poke 752,1: ? #6;"-----"
  _"? :? "PRESS START BUTTON TO START THE GAME":Poke 53279,0
22 ? "PRESS SELECT BUTTON FOR INSTRUCTIONS"
23 ? "ATARI COMPUTER ENTHUSIAST, EUGEN E. OR",
24 If Peek(53279)=5 Then Graphics 0:Go to 30
25 If Peek(53279)=6 Then 99
26 If Peek(53279)=5 Then Graphics 0:Go to 30
29 Goto 25
30 Poke 710,0: ? "  INSTRUCTIONS":?
  :? "THE OBJECT OF THE GAME IS TO CLIMB TO":? "THE TOP OF THE BUILDING.":?
31 ? "THIS IS NOT SO EASY AS YOU MUST DODGE":? "THE FALLING OBJECTS.":? "YOU CAN ONLY CLIMB UP ON THE WINDOWS."
32 ? :? "YOU MUST ALSO WATCH THAT YOU DON'T GO":? "OFF THE EDGE OF THE BUILDING":? :? "TO CONTROL YOUR MAN"
33 ? :? "MOVE THE JOYSTICK UP TO CLIMB":? :? "& JUST PUSH SIDEMAYS TO MOVE ACROSS"
34 ? "YOU SCORE 1 POINT PER FLOOR CLIMBED &":Poke 752,1
35 ? "YOU CAN'T GO DOWN.ONLY THE HARD WAY":? :? "YOU ARE ALLOWED 2 MISTAKES, THE THIRD":? "IS FATAL !!!!!"
40 ? :? "PRESS START BUTTON TO START THE GAME."
90 Poke 53279,0
95 If Peek(53279)<6 Then 95
99 Z=2
100 Graphics 21:Poke 559,46:Poke 752,1
102 Poke 704,222:Poke 705,142:Poke 706,188:Poke 707,30
104 Poke 708,6:Poke 709,130:Poke 710,130:Poke 712,0:Poke 711,6:Poke 623,17
106 Px=Peek(106)-16:Poke 54279,Px:Poke 53277,3
108 Poke 53256,0:Poke 53257,0:Poke 53258,0:Poke 53259,0:Poke 53260,255:Poke 53255,100
110 Poke 53248,130:Poke 53249,110:Poke 53250,130:Poke 53251,150:Poke 53252,120:Poke 53253,140:Poke 53254,160
120 Ox=256*Px+512-Ux:Bx=Int(Ox/256):Ax=Ox-256*Bx
121 Poke Vx+2,Ax:Poke Vx+3,Bx:Poke Vx+4,40:Poke Vx+5,2:Poke Vx+6,40:Poke Vx+7,2
190 Gosub 990:Gosub 900:F=0:Sc=0:C=2
195 P=28
198 I0=0.25:I1=-1:I2=-1:I3=1:H1=115:H2=130:H3=150:A=102:Hp=130
199 Poke 53248,Hp:P$(100,112)=A$:Sound 0,0,0,0:Ac=0:N=0:Gosub 340:P$(131,142)=X$
200 Poke 53278,0
210 St=Stick(0):If St=11 Or St=10 Or St=9 Then Sound 0,21,8,4:Hp=Hp-2:Poke 53248,Hp
211 If St=7 Or St=5 Or St=6 Then Sound 0,21,8,4:Hp=Hp+2:Poke 53248,Hp
215 O=Peek(53256)+Peek(53257)+Peek(53258)+Peek(53259)
220 If St=14 And U=0 And Int(O/2)=O/2 Then Sound 0,140,8,4:U=1:A=A-I0:P$(0,0+12)=B$:Gosub 310:Sc=Sc+1/3:Goto 230
221 If St=14 And U=1 And Int(O/2)=O/2 Then Sound 0,180,8,4:U=0:A=A-I0:P$(A-2,A+12)=A$:Gosub 310:Sc=Sc+1/3
230 If A<10 Then 360
270 Z$=P$(121,499)
271 P$(121+Z,499)=Z$:Sound 0,0,0,0
272 H1=H1+I1:Poke 53249,H1:If H1<100 Then I1=1:Gosub 350
273 If H1>160 Then I1=-1:Gosub 350
274 H2=H2+I2:Poke 53250,H2:If H2<100 Then I2=1:Gosub 350
275 If H2>160 Then I2=-1:Gosub 350
276 H3=H3+I3:Poke 53251,H3:If H3<100 Then I3=1:Gosub 350
277 If H3>160 Then I3=-1:Gosub 350
278 Sound 0,0,0,0
280 If Int(A)=60 And C=0 And I0<1 Then I0=2: For W=1 To 7:Sound 0,21,10,10: For Mw=1 To 7:Next Mw:Sound 0,0,0,0:Next M
285 If A/10=Int(A/10) And Ac=1 Then P=P-2:Ac=0
286 If P<15 Then P=15
288 If A/10<Int(A/10) Then Ac=1
290 If Hp>163 Or Hp<97 Or Peek(53260)<0 Then Goto 300
298 N=M+1:If N>P Then N=0:Gosub 340:P$(131,142)=X$
299 Goto 200
300 F=F+1:Poke 53249,1:Poke 53250,1:Poke 53251,1
301 For M=A To 115 Step 2:P$(M,M+12)=A$:C=C-0.5:If C<0 Then C=2
302 Sound 0,M,10,6:Sound 1,M,10,6:Gosub 330
304 Next M:Sound 0,123,8,14:Sound 1,0,0,0:P$(M,M+9)=D$: For Mw=1 To 14:Next Mw:Sound 0,0,0,0:If F=3 Then 380
305 For M=130 To 250 Step 5:P$(M,M+4)=♥♥♥♥♥:Next M:P$(244,254)=AM$:Poke 705,14:Poke 53257,1
306 For M=200 To 2 Step -1:Poke 53249,M:If M/10=Int(M/10) And M/4=Int(M/4) Then Sound 0,35,10,6:Goto 308
307 If W/10=Int(W/10) Then Sound 0,49,10,6
308 If Hp>M Then Poke 53248,1
309 Next M:Sound 0,0,0,0:Poke 705,14
310 If I0>1 Then Return
320 C=C+1:If C>2 Then C=0
330 If C=0 Then Poke 708,6:Poke 709,130:Poke 710,130
331 If C=1 Then Poke 708,130:Poke 709,6:Poke 710,130
332 If C=2 Then Poke 708,130:Poke 709,130:Poke 710,6
333 Return
340 Rn=Int(Rnd(0)*8+1):On Rn Goto 341,342,343,344,345,347,348,349
341 X$=S1$:Return
342 X$=S2$:Return
343 X$=H$:Return
344 X$=H$:Return
345 X$=S$:Return
347 X$=P1$:Return
348 X$="♥":Poke 77,0:Return
349 X$=E$:Return
350 Sound 0,Rnd(0)*20,10,4:Return
360 Restore 1010: For T=1 To 11:Read Fr:Read De:Sound 0,Fr,10,10: For W=1 To De:Next M:Sound 0,0,0,0:Next T
369 Z=Z+2:P=P-3:Gosub 370:Goto 198
370 For M=Px*256+620 To Px*256+1027:Poke M,0:Next M: For M=A To 494 Step 5:Sound 0,255-(M/2),8,4:P$(M,M+4)=♥♥♥♥♥
371 Next M:Sound 0,21,10,6:Return
380 For Mw=1 To 128:Poke 712,Mw:Sound 0,Mw,10,6:Sound 1,Mw-1,10,6:Poke 712,0:Next Mw
381 Sound 0,49,8,14:Sound 1,0,0,0:C=0:Gosub 330
382 P$(M,M+8)=G$: For M=140 To 0 Step -1:Sound 0,49,8,Int(M/10):Next M:Sound 0,0,0,0

```



# TYPEAHEAD by GREG MENKE

```

383 Position 0,29:? #6;"BBBBBBBBBBBBB
BBBBB"
384 ? #6;"BAAABAAAABAAAABAAAAB"? #6;"B
ABBBABABAAAABABBB"? #6;"BAAAAABABA
BABAAB"
385 ? #6;"BABABABABABABABBB"? #6;"B
AAAAABABABBBABAAAAB"
386 ? #6;"BBBBBBBBBBBBBBBBBB"? #6;"B
BAAAABABABAAAABABBB"? #6;"BBABABABABAB
BBABABBB"? #6;"BBABABABABABABBB"
387 ? #6;"BBABABAAAABABBBABBB"? #6;"B
BAAAABABABAAAABABBB"? #6;"BBBBBBBBBBBBB
BBBBBB"
388 For M=1 To 500:Next M
389 Poke 53248,1:Poke 53249,1:Poke 532
50,1:Poke 53251,1:Poke 53252,1
390 Graphics 18:? #6;"your score was "
;Int(Sc):? #6;"If Sc>High Then High=Int
(Sc)
391 ? #6;"HIGH SCORE IS ";High:? #6:?
#6:? #6:? #6;"Press start button TO
RESTART":Poke 53279,0
392 Poke 53253,1:Poke 53254,1:Poke 532
55,1
393 For M=1 To 50:Next M:Poke 53279,
0
394 If Peek(53279)<6 Then 394
395 Z=2:Goto 100
900 For C=1 To 3: For M=C-1 To C+44
Step 3:Color C:Plot 30,M:Drawto 35,M:
Plot 40,M:Drawto 45,M
901 Plot 50,M:Drawto 55,M
909 Next M:Next C:Return
990 For M=Px*256+384 To Px*256+1023:
Poke M,0:Next M
995 For M=Px*256+400 To Px*256+495:P
oke M,255:Next M
999 Return
1010 Data 100,35,120,20,90,17,100,35,1
20,20,0,15,90,15,90,15,80,16,80,16,75,
18

```



```

10 ;Atari TYPEAHEAD by Greg Menke.
20 ;
30 ;V1.0 4/29/84
40 ;
50 ;Uses Atari Assembler/Editor
60 ;format.
70 ;
80 ;
90 ;
0100 ;Raise NUM if the character
0110 ;printing is erratic.
0120 ;
0130 ;
0140 NUM = 2
0150 ;
0160 *=6000
0170 PLA
0180 JMP START
0190 ;
0200 LDA 12
0210 STA DOSINI+1
0220 LDA 13
0230 STA DOSINI+2
0240 JSR START
0250 RTS
0260 START LDA #RESTART&255
0270 STA 12
0280 LDA #RESTART/256
0290 STA 13
0300 JSR INIT
0310 RTS
0320 RESTART JSR START
0330 DOSINI JMP DOSINI
0340 ;
0350 INIT LDY #0
0360 MOVE LDA $E400,Y
0370 STA TABLE,Y
0380 CPY #16
0390 BEQ DONE
0400 INY
0410 JMP MOVE
0420 DONE CLC
0430 LDA $E404
0440 ADC #1
0450 STA EDITOR+1
0460 LDA $E405
0470 ADC #0
0480 STA EDITOR+2
0490 LDA #GET&255
0500 STA TABLE+4
0510 LDA #GET/256
0520 STA TABLE+5
0530 LDA #TABLE&255
0540 STA $321
0550 LDA #TABLE/256
0560 STA $322

```

```

0570 JMP CONT
0580 ;
0590 TABLE *=+20
0600 EDITOR JMP EDITOR
0610 FLAG .BYTE 0
0620 INDEX .BYTE 0
0630 INDEX2 .BYTE 0
0640 SKIP .BYTE 0
0650 ;
0660 ;Now initialize VBI routine
0670 ;
0680 CONT LDA #0
0690 STA INDEX
0700 STA INDEX2
0710 STA SKIP
0720 STA FLAG
0730 LDX #VBI/256
0740 LDY #VBI&255
0750 LDA #6
0760 JSR $E45C
0770 RTS
0780 ;
0790 ;Routines here
0800 ;
0810 VBI LDA FLAG
0820 CMP #0
0830 BNE WRITE
0840 ;Get a character from keyboard
0850 LDY INDEX
0860 CPY #BUFFER-BUFFER
0870 BEQ EXIT
0880 LDA 764
0890 CMP #255
0900 BEQ EXIT
0910 STA BUFFER,Y
0920 INC INDEX
0930 LDA #255
0940 STA 764
0950 JMP EXIT
0960 ;Send characters from BUFFER
0970 WRITE LDA SKIP
0980 CMP #0
0990 BEQ PASS
1000 DEC SKIP
1010 JMP EXIT
1020 PASS LDY INDEX2
1030 CPY INDEX
1040 BEQ RESET
1050 LDA BUFFER,Y
1060 STA 764
1070 INC INDEX2
1080 LDA #NUM
1090 STA SKIP
1100 EXIT JMP $E45F
1110 RESET LDA #0
1120 STA INDEX
1130 STA INDEX2

```



# ALPHABET BEE

by

```
; Alphabet Bee
; Stan Ockers 6-84
; Written in Action! (c) 1983 ACS
; ACE Newsletter, 3662 Vine Maple Dr.
; Eugene, OR 97405 June 1984 $12 year
;
MODULE
```

```
BYTE ARRAY dlist(72)=$0600,flwpos(108),
    flwchr(108)
BYTE j,pmpage,vpos=[100],scrnpage,
    hpos=[0],x,y,pickflg=[0],colr,chr,
    vpos2,sel,cnt,match=[65],p,q,
    dlicnt=$01,sndcnt1,sndflg1=[0],
    sndpos1=[0],maxloud=[13],audf1=$0200,
    audc1=$0201,falloff=[5],pospnt=[0],
    vcount=$0400,audf2=$0202,audc2=$0203,
    sndpos2=[0],sndflg2=[0],sndcnt2,pflg
CARD ptr=$60,pmbase,source,dest,
    scrnbase,csptr=[57344]
```

```
BYTE ARRAY beel=[0 0 216 80 112 248
    248 248 113 39 126 126 60 60 24 0
    0], bee2=[0 0 27 10 14 31 31 31
    142 228 126 126 60 60 24 0 0],
    wings1=[0 0 2 5 9 2 2 5 81 82 84
    0 2 42 40 20 20 0 0], wings2=[0 0
    64 160 144 64 64 160 138 74 42 0
    64 84 20 40 40 0 0],bee(16),
    wings(18),chr1=[0 0 0 0 0 0 0 0 0
    0],dli=[$48 $0A $48 $A6 $01 $B0
    $A8 $06 $B0 $A2 $06 $E8 $B0 $A8
    $06 $A8 $E8 $B0 $A8 $06 $B0 $A5
    $06 $E8 $B0 $A8 $06 $E8 $86 $D1
    $80 $0A $D4 $8C $00 $00 $80 $80
    $00 $68 $A0 $68 $40 $17 72 $18
    218 $16 32 $1A 212 $16 50 $1A 57],
    tune1(66),tune2(66),
    theme=[20 53 20 60 40 64
    10 72 10 64 10 60 10 72 40 81 10
    64 10 60 10 53 10 64 10 72 10 64
    10 60 10 72 10 64 10 60 10 53 10
    64 10 72 10 64 10 60 10 72 20 53
    20 60 40 64 10 72 10 64 10 60 10
    72 40 81 0 0],
    themeb=[10 128 10 108 10 144 10
    108 10 162 10 108 10 128 10 108 40
    121 10 128 10 108 10 128 10 108 20
    162 20 128 40 121 20 108 20 128 40
    121 10 128 10 108 10 144 10 108 10
    162 10 108 10 128 10 108 40 121 10
    128 10 108 20 162 0 0]
```

```
BYTE POINTER filit
PROC Arainit() ; set up display list
FOR j=0 TO 2 DO dlist(j)=112 00
FOR j=3 TO 60 STEP 3 DO
    dlist(j)=86 00 dlist(63)=70
    dlist(66)=134
```

```
FOR j=67 TO 68 DO dlist(j)=6 00
dlist(69)=65 dlist(70)=0 dlist(71)=6
dlist(64)=96 dlist(65)=Peek(106)-1
scrnpage=Peek(106)
FOR j=0 TO 18 STEP 2 DO
    dlist(3*j+5)=scrnpage+j/2
    dlist(3*j+8)=scrnpage+j/2
    dlist(3*j+4)=0
    dlist(3*j+7)=128 00 dlist(60)=$D6
    dlist(36)=$D6
RETURN
```

```
PROC Pminit() ; player Missiles
pmpage=Peek(106)+10 pmbase=pmpage*256
SetBlock(pmbase+1024,1024,0)
Poke(54279,pmpage) Poke(559,62)
Poke(53277,3) Poke(623,1)
Poke(53248,114) Poke(704,34)
Poke(705,46) Poke(53249,114)
Poke(53250,114) Poke(706,72)
Poke(53256,0) Poke(53257,0)
RETURN
```

```
PROC Csinic() ; a new character set
BYTE POINTER pick
BYTE ARRAY newchr=[8 12 12 14 14
    15 15 24 24 60 60 126 126 255 255
    16 16 48 48 112 112 240 240 15 15
    7 7 3 3 1 1 240 240 224 192 192
    128 128 255 255 126 126 60 60 24 24
    24 219 90 126 126 60 24 24 24 24
    24 24 24 24 24 0 0 31 63 127 255 255
    255 0 0 240 252 254 255 255 255 0 0
    252 240 240 192 0 0 0 63 15 15 3
    0 0 0 255 255 255 255 255 255]
source=csptr dest=pmbase
MoveBlock(dest,source,1024)
FOR pick=pmbase+256 TO pmbase+263
    DO pick^=255 00
FOR pick=pmbase+33*8 TO pmbase+59*8-1
    DO pick^=255 00
FOR j=0 TO 104
    DO pick=pmbase+24+j pick^=newchr(j) 00
    Poke(756,pmpage)
RETURN
```

```
PROC Dinit() ; insert new display list
FOR j=0 TO 58
    DO Poke($680+j,dli(j)) 00
Poke(512,$80) Poke(513,6) ptr=$600
Poke(54286,192)
RETURN
```

```
PROC Flwr(BYTE x,y,chr,colr) ; a flower
BYTE addon
addon=64*colr filit=scrnbase+128*y+x
filite=3+addon filite+=+1 filite=4+addon
```

```
filite+=+1 filite+=5+addon filite+=+126
filite+=6+addon filite+=+1
filite+=chr-32+addon filite+=+1
filite+=7+addon filite+=+127
filite+=8+addon filite+=+128 filite+=9
FOR j=y+4 TO 19
    DO filite+=128 filite+=10 00
RETURN
```

```
PROC Flwinit() ; a field of flowers
FOR j=0 TO 108 DO
    flwpos(j)=0 flwchr(j)=0 00
FOR j=65 TO 90 DO
    DO sel=Rand(98)+5 UNTIL
        flwpos(sel)=0 AND flwpos(sel-1)=0
        AND flwpos(sel+1)=0 00
    DO flwpos(sel)=6+3*Rand(4) UNTIL
        flwpos(sel)<flwpos(sel-2) AND
        flwpos(sel)<flwpos(sel+2) 00
    flwchr(sel)=j 00
FOR cnt=4 TO 103 DO
    IF flwpos(cnt)=0 AND flwpos(cnt+1)=0
        AND flwpos(cnt+2)=0 THEN
        DO flwpos(cnt+1)=6+3*Rand(4) UNTIL
            flwpos(cnt+1)<flwpos(cnt-1) AND
            flwpos(cnt+1)<flwpos(cnt+3) 00
        flwchr(cnt+1)=64 FI 00
FOR cnt=0 TO 108 DO
    IF flwpos(cnt)>0 THEN
        x=cnt+9 y=flwpos(cnt) colr=Rand(4)
        Flwr(x,y,flwchr(cnt),colr) FI 00
RETURN
```

```
PROC Hivepart(BYTE x,y,len)
filite=scrnbase+128*y+x filite+=11
FOR j=1 TO len-2 DO
    filite+=+1 filite+=15 00 filite+=+1
    filite+=12 filite+=+128-len
FOR j=1 TO len
    DO filite+=+1 filite+=32 00
RETURN
```

```
PROC Hivedoor(Byte x,y)
filite=scrnbase+128*y+x filite+=13
filite+=+1 filite+=0 filite+=+1
filite+=14 FOR j=1 TO 6
    DO filite+=+125 FOR cnt=1 TO 3
        DO filite+=+1 filite+=0 00 00
RETURN
```

```
PROC Hive(BYTE p,q) ; draw a hive
Hivepart(p+3,q,3) Hivepart(p+2,q+2,5)
Hivepart(p+1,q+4,7) FOR CNT=0 TO 5
    DO Hivepart(p,q+6+2*cnt,9) 00
RETURN
```

```
PROC Scrninit() ; draw background
```



# STAN OCKERS

```

CARD num
scrnbase=scrnpage*256 Poke(712,144)
SetBlock(scrnbase,2560,0)
Poke(709,54) Poke(710,102) Poke(711,68)
Flwinit()
Poke(656,1) Poke(657,0)
Hive(5,2) Hivedoor(10,10)
Hive(118,2) Hivedoor(120,10)
RETURN

PROC Plapart() ; play part of theme
cnt=0 j=0 DO
  tune1(j)=theme(pospnt)
  IF j MOD 2=0 THEN cnt==+theme(pospnt)
  FI j==+1 pospnt==+1
  UNTIL cnt=80 DO tune1(j)=theme(pospnt)
  tune1(j+1)=0 pospnt==+1
  IF pospnt=64 THEN pospnt=0 FI
  audf1=tune1(1) sndcnt1=tune1(0)
  sndpos1=1 sndflg1=1 audc1=$A0%maxloud
RETURN

PROC Plall() ; play all of theme
FOR j=0 TO 65 DO tune1(j)=theme(j) DO
  audf1=theme(1) sndcnt1=theme(0)
  sndpos1=1 sndflg1=1 audc1=$A0%maxloud
  FOR j=0 TO 65 DO tune2(j)=themeb(j) DO
    audf2=themeb(1) sndcnt2=themeb(0)
    sndpos2=1 sndflg2=1 audc2=$A0%maxloud
  RETURN

PROC Grab() ; take a letter
x=hpos y=flwpos(x) chr=flwchr(x)
filit=scrnbase+128*y+x+9*colr=filit+
colr=-/64 vpos2=vpos+16
Flwr(x+9,y,64,colr) flwchr(x)=64
pickflg=1 source=csptr+(chr-32)*8
dest=pmbase+1536+vpos2
MoveBlock(dest,source,8) dest=chr+1
MoveBlock(dest,source,8)
Plapart() Poke(77,0)
RETURN

PROC Letgo() ; drop a letter
FOR j=1 TO 8 DO chr1(j)=0 DO
  source=chr1 dest=pmbase+1536+vpos2
  MoveBlock(dest,source,10)
  pickflg=0
  IF chr(>)match THEN ; back if no match
  DO sel=Rand(98)+5 UNTIL
    flwchr(sel)=64 DO
    flwchr(sel)=chr x=sel y=flwpos(sel)
    colr=Rand(4) flwr(x+9,y,chr,colr)
    ELSE Put(chr) match==+1
    IF match=91 AND pflg=0 THEN Plall()
    pflg=1 ELSE Plapart() FI FI
RETURN

PROC Abet() ; show/remove alphabet
BYTE flipit=[0]
filit=(scrnpage-1)*256+96 Poke(54286,64)
FOR j=161 TO 186 DO
  IF flipit=0 THEN filit^=j ELSE
    filit^=0 FI filit==+1 DO
  IF flipit=0 THEN flipit=1 ELSE
    flipit=0 FI
  DO UNTIL Peek(53279)<>3 DO Poke(54286,194)
RETURN

PROC Main()
BYTE byt,stk=$278
INT hscr=[0]
CARD wait
j=Peek(106) Poke(106,j-18)
DO FOR j=0 TO 16 DO bee(j)=bee1(j) DO
  FOR j=0 TO 18 DO wings(j)=wings1(j) DO
  Arainit() Graphics(1) Pminit()
  Csinit() Scrninit() Dlininit() Plall()
  source=(Peek(106)-1)*256
  Setblock(source,256,0)
  match=65 hpos=0 vpos=100 pflg=0
  DO
  DO UNTIL vcount=128 DO dlicnt=0
  IF sndflg1=1 THEN sndcnt1=-1
  IF sndcnt1<falloff AND sndcnt1>0
    THEN audc1=-1 FI
  IF sndcnt1=0 THEN sndpos1==+1
  sndcnt1=tune1(sndpos1)
  IF sndcnt1=0 THEN sndflg1=0
  audc1=160 ELSE sndpos1==+1
  audf1=tune1(sndpos1)
  audc1=$A0 % maxloud FI FI FI
  IF sndflg2=1 THEN sndcnt2=-1
  IF sndcnt2<falloff AND sndcnt2>0
    THEN audc2=-1 FI
  IF sndcnt2=0 THEN sndpos2==+1
  sndcnt2=tune2(sndpos2)
  IF sndcnt2=0 THEN sndflg2=0
  audc2=160 ELSE sndpos2==+1
  audf2=tune2(sndpos2)
  audc2=$A0 % maxloud FI FI FI
  IF (stk&4)=0 AND hpos>0 THEN hscr==+1
  IF hscr=8 THEN FOR j=4 TO 61 STEP 3
  DO dlist(j)=-1 DO hscr=0 hpos=-1 FI
  FOR j=0 TO 16 DO bee(j)=bee1(j) DO
  FOR j=0 TO 18 DO wings(j)=wings1(j) DO
  Poke(54276,hscr) FI
  IF (stk&8)=0 AND hpos<111 THEN hscr=-1
  IF hscr=-1 THEN FOR j=4 TO 61 STEP 3
  DO dlist(j)=+1 DO hscr=7 hpos==+1 FI
  FOR j=0 TO 16 DO bee(j)=bee2(j) DO
  FOR j=0 TO 18 DO wings(j)=wings2(j) DO
  Poke(54276,hscr) FI
  IF (stk&1)=0 AND vpos>20 THEN
    vpos=-1 vpos2=-1 FI
  IF (stk&2)=0 AND vpos<170 THEN
    vpos==+1 vpos2==+1 FI
  IF Rand(16)<1 THEN vpos==+1
    vpos2==+1 FI
  IF Rand(16)<1 THEN vpos=-1
    vpos2=-1 FI
  dest=pmbase+1021+vpos
  source=wings MoveBlock(dest,source,19)
  dest=pmbase+1280+vpos source=bee
  MoveBlock(dest,source,17)
  dest=pmbase+1536+vpos2 source=chr1
  MoveBlock(dest,source,10)
  IF Strig(0)=0 AND pickflg=0 AND
    flwchr(hpos)>64 AND flwchr(hpos)<91
    AND vpos>flwpos(hpos)*8+20
    AND vpos<flwpos(hpos)*8+30
    THEN Grab() FI
  IF Strig(0)=0 AND pickflg=1 AND
    (hpos=0 OR hpos=111) THEN Letgo() FI
  IF Peek(53279)=3 THEN Abet() FI
  IF Peek(53279)=6 THEN Poke(54286,64)
  EXIT FI DO DO
RETURN

```





# MX-80 Graphics Dump by GREG MENKE

```

1 Rem *****
2 Rem ** ACE NEWSLETTER **
3 Rem ** 3662 VINE MAPLE DR **
4 Rem ** EUGENE, OR 97405 **
5 Rem ** MAY 1984 $12 YEAR **
6 Rem *****
20 Rem MX-80 GRAPHICS DUMPER V2.0**
22 Rem * BY GREG MENKE 3/4/84 **
30 Rem *****
100 Dim L$(480),A$(20),F$(20),T$(120),
    Sc$(1),C$(8)
110 Graphics 0:SetColor 2,12,2
120 X=Fre(0)-4096:Dim Buff$(X)
130 Open #1,6,0,"D:*.*":Trap 150
140 Input #1;A$;? A$;:Input #1;A$;? A$
    :Goto 140
150 Close #1;? :? :? :? "Enter file to
    be printed. (You have"? X;" free byt
    es, ";Int(X/125);" sectors.)"
160 ? :Input F$:If Len(F$)<2 Then 130
170 ? "Enter TITLE. RETURN only if no
    ne."?:Input T$
180 ? "When printing, should I turn o
    ff the screen? (Y/N) "?:Input Sc$
185 If Sc$<"Y" And Sc$<"N" Then 180
190 ? "Turn on your Printer/Interface
    , then hit SELECT....."
200 If Peek(53279)<6 Then 200
210 Open #2,8,0,"P":If Len(T$)>0 Then
    ? #2;? #2;T$;? #2;? #2;? #2;? #2
220 ? "Press SELECT to load "F$;? :?
    :?
230 If Peek(53279)<5 Then 230
240 Open #1,4,0,F$:Trap 260
250 Get #1,A:?"%";Chr$(A);:Buff$(Len(
    Buff$)+1)=Chr$(A):Goto 250
260 ? "Printing "F$;? :? :?
400 Rem
410 Rem
420 If Sc$="Y" Then Poke 559,0
430 ? #2;Chr$(27);"P";? #2;Ch=1:L$=""
435 Rem
437 Rem
440 If Ch>Len(Buff$) Then GOSUB 600:Go
    to 640
445 For G=1 To 60
450 A=Asc(Buff$(Ch,Ch)):Ch=Ch+1:If C
    h>Len(Buff$) Then GOSUB 600:Goto 640
470 If Sc$="N" Then ? "%";Chr$(A);
480 If A=155 Then GOSUB 600:? #2;Got

```

```

0 440
490 Rem
520 Restore 1000+A
530 Rem
540 For H=1 To 8:Read A:L$(Len(L$)
    +1)=Chr$(A):Next H:Next G
550 GOSUB 600:Goto 440
560 Rem
585 End
600 X=Len(L$):Y=0:If X>255 Then Y=1:X=
    X-256
610 ? #2;Chr$(27);"K";Chr$(X);Chr$(Y);
620 For G=1 To Len(L$):? #2;L$(G,G);
    :Next G:L$="":Return
630 Rem
631 End
640 Close #2:LPrint Chr$(27);"Q":LPrin
    t Chr$(12)
650 Graphics 0:Setcolor 2,12,2:Poke 77
    ,0
660 ? :? "File "F$;" printed!"
670 ? :? :? :? "Press....."
680 ? :? "SELECT : RUN again."?: "SELECT
    " : End."
690 If Peek(53279)=6 Then Clr :Run
700 If Peek(53279)<5 Then 690
710 ? :Clr :End
1000 Data 0,48,120,124,62,124,120,48
1001 Data 0,0,0,255,255,24,24,24
1002 Data 0,0,0,0,0,0,255,255
1003 Data 24,24,24,248,248,0,0,0
1004 Data 24,24,24,255,255,0,0,0
1005 Data 24,24,24,31,31,0,0,0
1006 Data 3,7,14,28,56,112,224,192
1007 Data 192,224,112,56,28,14,7,3
1008 Data 1,3,7,15,31,63,127,255
1009 Data 0,0,0,0,15,15,15,15
1010 Data 255,127,63,31,15,7,3,1
1011 Data 0,0,0,0,240,240,240,240
1012 Data 240,240,240,240,0,0,0,0
1013 Data 192,192,192,192,192,192,192,
    192
1014 Data 3,3,3,3,3,3,3,3
1015 Data 15,15,15,15,0,0,0,0
1016 Data 0,24,24,122,102,122,24,24
1017 Data 0,0,0,31,31,24,24,24
1018 Data 24,24,24,24,24,24,24,24
1019 Data 24,24,24,255,255,24,24,24
1020 Data 0,28,62,62,62,62,28,0
1021 Data 15,15,15,15,15,15,15,15
1022 Data 255,255,0,0,0,0,0,0
1023 Data 24,24,24,31,31,24,24,24
1024 Data 24,24,24,248,248,24,24,24
1025 Data 255,255,255,255,0,0,0,0
1026 Data 0,0,0,248,248,24,24,24
1027 Data 0,248,248,174,174,10,10,0
1028 Data 0,16,48,126,126,48,16,0
1029 Data 0,8,12,126,126,12,8,0
1030 Data 0,16,56,124,84,16,16,0

```

```

1031 Data 0,16,16,84,124,56,16,0
1032 Data 0,0,0,0,0,0,0,0
1033 Data 0,0,0,122,122,0,0,0
1034 Data 0,112,112,0,0,112,112,0
1035 Data 36,126,126,36,36,126,126,36
1036 Data 0,36,116,214,214,92,72,0
1037 Data 0,102,108,24,48,102,70,0
1038 Data 0,12,94,242,186,236,78,10
1039 Data 0,0,0,112,112,0,0,0
1040 Data 0,0,0,60,126,102,66,0
1041 Data 0,66,102,126,60,0,0,0
1042 Data 16,84,124,56,56,124,84,16
1043 Data 0,16,16,124,124,16,16,0
1044 Data 0,0,1,7,6,0,0,0
1045 Data 0,16,16,16,16,16,16,0
1046 Data 0,0,0,6,6,0,0,0
1047 Data 0,6,12,24,48,96,64,0
1048 Data 0,60,126,74,82,126,60,0
1049 Data 0,2,34,126,126,2,2,0
1050 Data 0,34,102,78,90,114,34,0
1051 Data 0,68,70,82,122,110,68,0
1052 Data 0,12,28,52,126,126,4,0
1053 Data 0,116,118,82,82,94,76,0
1054 Data 0,60,126,82,82,94,12,0
1055 Data 0,64,70,78,88,112,96,0
1056 Data 0,44,126,82,82,126,44,0
1057 Data 0,32,114,82,86,124,56,0
1058 Data 0,0,0,54,54,0,0,0
1059 Data 0,0,1,55,54,0,0,0
1060 Data 0,0,16,56,108,198,130,0
1061 Data 0,36,36,36,36,36,36,0
1062 Data 0,130,198,108,56,16,0,0
1063 Data 0,32,96,74,90,112,32,0
1064 Data 0,60,126,66,90,122,58,0
1065 Data 0,30,62,100,100,62,30,0
1066 Data 0,126,126,82,82,126,44,0
1067 Data 0,60,126,66,66,102,36,0
1068 Data 0,126,126,66,102,60,24,0
1069 Data 0,126,126,82,82,82,66,0
1070 Data 0,126,126,80,80,80,64,0
1071 Data 0,60,126,66,74,78,78,0
1072 Data 0,126,126,16,16,126,126,0
1073 Data 0,66,66,126,126,66,66,0
1074 Data 0,4,6,2,2,126,124,0
1075 Data 0,126,126,24,60,102,66,0
1076 Data 0,126,126,2,2,2,2,0
1077 Data 0,126,126,48,24,48,126,126
1078 Data 0,126,126,56,28,126,126,0
1079 Data 0,60,126,66,66,126,60,0
1080 Data 0,126,126,72,72,120,48,0
1081 Data 0,60,126,66,68,126,58,0
1082 Data 0,126,126,72,76,126,50,0
1083 Data 0,32,114,82,82,94,12,0
1084 Data 0,64,64,126,126,64,64,0
1085 Data 0,126,126,2,2,126,126,0
1086 Data 0,120,124,6,6,124,120,0
1087 Data 0,126,126,12,24,12,126,126
1088 Data 0,102,126,24,24,126,102,0
1089 Data 0,96,112,30,30,112,96,0

```



1090 Data 0,70,78,90,114,98,66,0  
 1091 Data 0,0,0,126,126,66,66,0  
 1092 Data 0,96,48,24,12,6,2,0  
 1093 Data 0,66,66,126,126,0,0,0  
 1094 Data 0,8,24,48,96,48,24,0  
 1095 Data 2,2,2,2,2,2,2,2  
 1096 Data 0,24,60,126,126,60,24,0  
 1097 Data 0,4,46,42,42,62,30,0  
 1098 Data 0,126,126,18,18,30,12,0  
 1099 Data 0,28,62,34,34,34,0,0  
 1100 Data 0,12,30,18,18,126,126,0  
 1101 Data 0,28,62,42,42,58,24,0  
 1102 Data 0,0,16,62,126,80,80,0  
 1103 Data 0,25,61,37,37,63,62,0  
 1104 Data 0,126,126,16,16,30,14,0  
 1105 Data 0,0,18,94,94,2,0,0  
 1106 Data 0,0,1,1,1,95,94,0  
 1107 Data 0,126,126,8,28,22,2,0  
 1108 Data 0,0,66,126,126,2,0,0  
 1109 Data 0,62,62,24,28,56,62,30  
 1110 Data 0,62,62,32,32,62,30,0  
 1111 Data 0,28,62,34,34,62,28,0  
 1112 Data 0,63,63,36,36,60,24,0  
 1113 Data 0,24,60,36,36,63,63,0  
 1114 Data 0,62,62,32,32,48,16,0  
 1115 Data 0,18,58,42,42,46,36,0  
 1116 Data 0,32,32,124,126,34,34,0  
 1117 Data 0,60,62,2,2,62,62,0  
 1118 Data 0,56,60,6,6,60,56,0  
 1119 Data 0,56,62,14,28,14,62,56  
 1120 Data 0,34,54,28,28,54,34,0  
 1121 Data 0,57,61,5,7,62,60,0  
 1122 Data 0,34,38,46,58,50,34,0  
 1123 Data 0,24,58,126,126,58,24,0  
 1124 Data 0,0,0,255,255,0,0,0  
 1125 Data 0,124,124,112,120,94,78,0  
 1126 Data 0,16,56,124,254,0,0,0  
 1127 Data 0,0,0,254,124,56,16,0  
 1128 Data 255,207,135,131,193,131,135,207  
 1129 Data 255,255,255,0,0,231,231,231  
 1130 Data 255,255,255,255,255,255,0,0  
 1131 Data 231,231,231,7,7,255,255,255  
 1132 Data 231,231,231,0,0,255,255,255  
 1133 Data 231,231,231,224,224,255,255,255  
 1134 Data 252,248,241,227,199,143,31,63  
 1135 Data 63,31,143,199,227,241,248,252  
 1136 Data 254,252,248,240,224,192,128,0  
 1137 Data 255,255,255,255,240,240,240,240  
 1138 Data 0,128,192,224,240,248,252,254  
 1139 Data 255,255,255,255,15,15,15,15  
 1140 Data 15,15,15,15,255,255,255,255  
 1141 Data 63,63,63,63,63,63,63,63

1205 Data 255,129,129,207,231,207,129,129  
 1206 Data 255,129,129,199,227,129,129,255  
 1207 Data 255,195,129,189,189,129,195,255  
 1208 Data 255,129,129,183,183,135,207,255  
 1209 Data 255,195,129,189,187,129,197,255  
 1210 Data 255,129,129,183,179,129,205,255  
 1211 Data 255,223,141,173,173,161,243,255  
 1212 Data 255,191,191,129,129,191,191,255  
 1213 Data 255,129,129,253,253,129,129,255  
 1214 Data 255,135,131,249,249,131,135,255  
 1215 Data 255,129,129,243,231,243,129,129  
 1216 Data 255,153,129,231,231,129,153,255  
 1217 Data 255,159,143,225,225,143,159,255  
 1218 Data 255,185,177,165,141,157,189,255  
 1219 Data 255,255,255,129,129,189,189,255  
 1220 Data 255,159,207,231,243,249,253,255  
 1221 Data 255,189,189,129,129,255,255,255  
 1222 Data 255,247,231,207,159,207,231,247  
 1223 Data 253,253,253,253,253,253,253,253  
 1224 Data 255,231,195,129,129,195,231,255  
 1225 Data 255,251,209,213,213,193,225,255  
 1226 Data 255,129,129,237,237,225,243,255  
 1227 Data 255,227,193,221,221,221,255,255  
 1228 Data 255,243,225,237,237,129,129,255  
 1229 Data 255,227,193,213,213,197,231,255  
 1230 Data 255,255,239,193,129,175,175,255  
 1231 Data 255,230,194,218,218,192,193,255  
 1232 Data 255,129,129,239,239,225,241,255  
 1233 Data 255,255,237,161,161,253,255,255

1142 Data 252,252,252,252,252,252,252,252  
 1143 Data 240,240,240,240,255,255,255,255  
 1144 Data 255,231,231,133,153,133,231,231  
 1145 Data 255,255,255,224,224,231,231,231  
 1146 Data 231,231,231,231,231,231,231,231  
 1147 Data 231,231,231,0,0,231,231,231  
 1148 Data 255,227,193,193,193,193,227,255  
 1149 Data 240,240,240,240,240,240,240,240  
 1150 Data 0,0,255,255,255,255,255,255  
 1151 Data 231,231,231,224,224,231,231,231  
 1152 Data 231,231,231,7,7,231,231,231,231  
 1153 Data 0,0,0,0,255,255,255,255  
 1154 Data 255,255,255,7,7,231,231,231  
 1155 Data 255,7,7,81,81,245,245,255  
 1156 Data 255,239,207,129,129,207,239,255  
 1157 Data 255,247,243,129,129,243,247,255  
 1158 Data 255,239,199,131,171,239,239,255  
 1159 Data 255,239,239,171,131,199,239,255  
 1160 Data 255,255,255,255,255,255,255,255  
 1161 Data 255,255,255,133,133,255,255,255  
 1162 Data 255,143,143,255,255,143,143,255  
 1163 Data 219,129,129,219,219,129,129,219  
 1164 Data 255,219,139,41,41,163,183,255  
 1165 Data 255,153,147,231,207,153,185,255  
 1166 Data 255,243,161,13,69,19,177,245  
 1167 Data 255,255,255,143,143,255,255,255  
 1168 Data 255,255,255,195,129,153,189,255  
 1169 Data 255,189,153,129,195,255,255,255  
 1170 Data 239,171,131,199,199,131,171,239  
 1171 Data 255,239,239,131,131,239,239,255  
 1172 Data 255,255,254,248,249,255,255,255  
 1173 Data 255,239,239,239,239,239,239,239  
 1174 Data 255,255,255,249,249,255,255,255



```

1175 Data 255,249,243,231,207,159,191,
255
1176 Data 255,195,129,181,173,129,195,
255
1177 Data 255,253,221,129,129,253,253,
255
1178 Data 255,221,153,177,165,141,221,
255
1179 Data 255,187,185,173,133,145,187,
255
1180 Data 255,243,227,203,129,129,251,
255
1181 Data 255,139,137,173,173,161,179,
255
1182 Data 255,195,129,173,173,161,243,
255
1183 Data 255,191,185,177,167,143,159,
255
1184 Data 255,211,129,173,173,129,211,
255
1185 Data 255,223,141,173,169,131,199,
255
1186 Data 255,255,255,201,201,255,255,
255
1187 Data 255,255,254,200,201,255,255,
255
1188 Data 255,255,239,199,147,57,125,2
55
1189 Data 255,219,219,219,219,219,219,
255
1190 Data 255,125,57,147,199,239,255,2
55
1191 Data 255,223,159,181,165,143,223,
255
1192 Data 255,195,129,189,165,133,197,
255
1193 Data 255,231,195,153,153,129,153,
255
1194 Data 255,129,129,173,173,129,211,
255
1195 Data 255,195,129,189,189,153,219,
255
1196 Data 255,129,129,189,153,195,231,
255
1197 Data 255,129,129,173,173,173,189,
255
1198 Data 255,129,129,175,175,175,191,
255
1199 Data 255,195,129,189,181,177,177,
255
1200 Data 255,129,129,239,239,129,129,
255
1201 Data 255,189,189,129,129,189,189,
255
1202 Data 255,251,249,253,253,129,131,
255
1203 Data 255,129,129,231,195,153,189,
255
1204 Data 255,129,129,253,253,253,253,
255

```

```

34 Data 255,255,254,254,254,160,161,
5
1235 Data 255,129,129,247,227,233,253,
255
1236 Data 255,255,189,129,129,253,255,
255
1237 Data 255,193,193,231,227,199,193,
225
1238 Data 255,193,193,223,223,193,225,
255
1239 Data 255,227,193,221,221,193,227,
255
1240 Data 255,192,192,219,219,195,231,
255
1241 Data 255,231,195,219,219,192,192,
255
1242 Data 255,193,193,223,223,207,239,
255
1243 Data 255,237,197,213,213,209,219,
255
1244 Data 255,223,223,131,129,221,221,
255
1245 Data 255,195,193,253,253,193,193,
255
1246 Data 255,199,195,249,249,195,199,
255
1247 Data 255,199,193,241,227,241,193,
199
1248 Data 255,221,201,227,227,201,221,
255
1249 Data 255,198,194,250,248,193,195,
255
1250 Data 255,221,217,209,197,205,221,
255
1251 Data 255,231,197,129,129,197,231,
255
1252 Data 255,255,255,0,0,255,255,255
1253 Data 255,131,131,143,135,161,177,
255
1254 Data 255,239,199,131,1,255,255,25
5
1255 Data 255,255,255,1,131,199,239,25
5

```

```

0 REM *****
1 REM * FLASHING NUMBERS *
2 REM * WRITTEN BY *
3 REM * CHRIS SIMERAL *
4 REM * 54 E. WHITE TERRACE *
5 REM * NUTLEY, NJ 07110 *
6 REM *****
10 DIM A$(1)
13 DIM N(20)
14 ? NR;" OUT OF ";NG;" CONSEC. RIGHT
-";CR;" ? " PRESS RETURN TO START";:INPU
T A$
15 GRAPHICS 0
20 A=INT(999999*RND(1)+1)
21 REM TO MAKE A LARGER OR SMALLER NUM
BER CHANGE THE NUMBER OF 9'S AND 1'S I
N LINES 20 AND 30
30 IF A<111111 THEN 20
40 NG=NG+1;FOR X=1 TO 60:NEXT X:POSITI
ON 9,10;" " ;A
41 REM TO SPEED UP THE FLASH CHANGE LI
NE 40 TO READ FOR X=1 TO 50 OR 40 ETC.
50 FOR I=1 TO 160:NEXT I:GOTO 60
60 GRAPHICS 0;? "ENTER THE NUMBER YOU
SAW";:TRAP 60:INPUT N:TRAP 40000
65 IF N=A THEN ? "RIGHT":CR=CR+1:NR=NR
+1:GOTO 14
70 IF N<>A THEN ? "WRONG. IT WAS ";A:
CR=0:GOTO 14
75 GOTO 60

```

TYPEAHEAD by  
MENKE con't

```

1140 JMP EXIT
1150 ;
1160 ;Get routine here
1170 ;
1180 GET NOP
1190 PHA
1200 LDA #1
1210 STA FLAG
1220 PLA
1230 JSR EDITOR
1240 PHA
1250 LDA #0
1260 STA FLAG
1270 PLA
1280 RTS
1290 BUFFER = 1021
1300 BUFEND = 1151

```





TYING SOME LOOSE ENDS  
by Ruth Ellsworth

The ability to use strings in a computer language makes it possible to do all sorts of things with words or combinations of words and numbers. In PILOT these abilities are increased and easily used through the use of the MS: or Match String command.

Simple string variables are created by assigning a variable name after the \$ symbol. These storage files or shelves can then be called and used within a computer program (see the Dec/Jan. 1984 ACE NEWSLETTER).

"Concatenating" (combining) strings in PILOT makes it possible to create new string from two or more strings. For example, in a program to make compound words, C:\$NEWWORD=\$WORDONE\$WORDTWO would create a single compound word.

Associated string values make it possible to pass values in PILOT from one string to another (see the Dec/Jan. 1983 ACE NEWSLETTER). Thus, \$ONE=TWO and \$TWO=THREE will make \$ONE=THREE.

Probably the most powerful string tool is the MS: or Match String command in PILOT. Using the MS: command one can break a string down and use the parts of a string in a program. The MS: command breaks the string into three parts; \$LEFT, \$MATCH, and \$RIGHT. The RHYME TYME program at the end of this article illustrates the simple use of the MS: command. Line 60 of this program uses the MS: command to match the vowels in \$WORD. If one were to stop the program at that point and type T:\$LEFT,\$RIGHT,\$MATCH; the vowel matched would be \$MATCH, the consonants before the vowel would appear as \$LEFT, and the remainder of \$WORD would appear as \$RIGHT.

I have only used \*TEST twice in this program because we found that most of the words the children were using only needed to be checked for vowels twice to produce a good morphograph for rhyming. The \*TEST module could be repeated as often as desired but will not produce the desired result if two vowels appear together because only the last vowel matched becomes \$MATCH as the program is written. (The technique used to peel the letter off a string one at a time will be discussed in a separate article.)

In order to use \$RIGHT in \*TEST, \$RIGHT must be placed in the internal accept buffer with the A:= command. \$RIGHT is then matched. If \$RIGHT has no match there was no vowel in \$RIGHT, and \$MORPH becomes \$FIRST. \$MORPH is created by concatenating \$MATCH\$RIGHT from the last \$RIGHT that contains a vowel after two matches.

Parsing (separating) strings is easily done in PILOT through the use of the MS: command if one places a separator in the string matched. Any character can be used as the separator. I have chosen to use a comma in the SAVECHR module at the end of this article because commas have to be inserted to save a list of strings under a filename. The SAVECHR module may be substituted for the WRITE:D: lines in last month's CHARACTER MAKER, and the RETRCHR module may be substituted for the CHARACTER RETRIEVAL ROUTINE. The advantage to using SAVECHR is that one may save any character generated by the CHARACTER MAKER under any chosen name. Be sure to type D:filename. D: could have been placed in the program as a string, such as \$D=D: and then matched, but would have added unnecessary length to the listing. Because it is possible in PILOT to pass numerical values to string, the values generated in the CHARACTER MAKER are converted into strings so that they may be saved and easily retrieved under \$FILENAME.

RETRCHR routine reads the string values saved under \$FILENAME (\$STRING). \$STRING is then placed in the accept buffer to be used in the program. The comma is matched in Line 280 by using the vertical line before the comma. Line 70 could have read C:\$A=#A,\$B=#B, etc., but my machine freezes when RETRCHR is run with that listing. The values are received and usable after system reset is used. If you are very short on K or really hate typing, you may try that listing before you run the program in which the values are to be used.

\$STRING is matched to the first comma, the value of \$RIGHT is passed to \$STRING in Line 370 and the match is repeated. A counter keeps track of each time the string is matched and each \$LEFT is assigned a numeric variable according to the number of the counter. Though ecologically unsound, one could think of the process as being like stripping leaves off a tree branch. As each leaf is removed it is assigned a numeric variable and discarded while the remaining leaves are renamed (\$STRING). When the last leaf remains, it is left in the \$RIGHT file and given a numeric variable. In this case we knew how many string variables we were going to encounter and could use a counter to determine the end of our routine. If the number of variables is not known, one could use non print characters to match the blanks which surround strings in PILOT. The important thing to remember is using this technique is to keep the part of the string to be used the same throughout the routine. If one is discarding \$LEFT with each pass, one must rename \$RIGHT and reuse it unless one wants undesirable results



## ERACE DISK UPDATE

Erace disks 6 and 7 are now ready. We want to give special thanks to John R. Kelley for his contributions. All of disk 6 is a collection of programs he has sent us. We also want to thank the members of the New South Wales group who sent us some of their work. The first seven programs of disk 7 are from them.

Following is a list of the programs on the two disks.

### ERACE DISK #6

**WORLD CAP**—A text quiz asking you to name the capitals of the countries of the world.

**FRACTION**—Provide two fractions and the program will add, subtract, multiply or divide them and show you the reduced form.

**HANGMAN.REV**—An expanded hangman game with 310 words included.

**GEOCENTE**—State Geographic Centers. See how many states you can identify by the cities or towns near their geographic centers.

**DOMINOS**—Solve arithmetic problems based on the spots on the dominos. This drill is good for 6-9 year olds.

**DICEMATH**—This drill uses dice to set up arithmetic problems. It is similar to Dominos, but the problems are easier.

**YATCC**—A game of chance to help teach logic patterns. It is based on the dice game Yahtzee.

**SCRIPTURE**—A quiz to test your knowledge of biblical events, places and persons.

**ALGEDRIL**—Drills and timed tests on basic algebra. There are four different types of problems.

**AREAS.REV**—This program will give the area or volume of a chosen shape when you give the needed dimensions.

### ERACE DISK #7

**SUMS.BAS**—A simple program for addition, subtraction and multiplication.

**POLYGON.BAS**—draws polygons based on the number of sides and side length the user provides.

The next six math programs use the same format for presenting problems. Each will guide you through the type of problem indicated by the title of the program, correcting and rewarding you as you work each step of a problem. These are very good programs for students learning how to work a particular type of problem. There are several levels in each program.

**DIVISION.BAS**

**MULTIPLY.BAS**

**SUMS2.BAS**—Addition.

**IMPFRAC.BAS**—Improper fractions.

**EQUFRAC.BAS**—Equivalent fractions.

**ADDFRAC.BAS**—Add and reduce fractions.

**DICEPROB.ACE**—Work out probability tables on dice throws using 2 die. Has printer option.

**DICEROLL**—Use 2 or 3 die to produce probability tables. It is similar to the above program, but produces a distribution table when it is done.

**ABSTRACT**—Given clues by the computer, you must determine an unknown 3 digit number. Similar to master mind. Good logic builder.

## TARGETS

### A NUMBER GAME

**TARGETS** (Sunburst Communications, \$40) is a number game for ages 8 to adult. It designed to provide practice and creative application of basic math facts. This program is not designed to teach basic math facts, and could easily frustrate an individual not fluent in either addition, subtraction or multiplication and division.

Game options do allow a child to perform only addition, or only addition and subtraction, or a combination of addition (Level I), subtraction (Level II), multiplication and division (III). Clear, concise instructions included on the program allow a child to easily choose one of the above math functions. Then, the child is given the choice of either choosing the target number from 1-99 (the answer), or choosing two numbers from 1 - 9 and having the computer select the target. Next the child must combine the two numbers (utilizing the selected operations) to reach the target number. Thus, a child can practice basic operations.

Immediate feedback is given regarding how many steps it takes to reach a target. This becomes a challenge to create the shortest steps to achieve a target. Because one is able to erase the last number or sign entered, or restart the problem, it is possible to test out possible solutions. In some cases it is impossible to reach the target with the given combination of numbers. A response of "impossible" is also an option.

There are also 30 championship games at each of the 3 program levels. The computer maintains an ongoing record of the name and number of steps of the "champion" of each game.

If a child solves the problem in fewer steps than the champion of record, she or he becomes the new champion. The teacher has the option of having the computer display, or not display the solutions to championship problems.

This program is flexible enough for Miles, a six-year-old, to use. We selected low target numbers, and used only the addition and subtraction levels. Further, it is designed to provide stimulating, creative challenge to the student who has mastered basic computation facts. This should not be considered a game, but rather, a mind-expanding program requiring concentrated application of basic math skills combined with systematic goal-directed problem solving.

—Alice Miles Erickson Miles Erickson



## MODE MIXER

**MODE MIXER** (Xlent Software, Box 5228, Springfield, VA 22150, \$20) contains three programs for beginners and advanced programmers who want to mix graphics modes on the Atari screen. The opening menu presents 4 choices: 1.) Mode Mixer 1; 2.) M M 1 Demo; 3.) Mode Mixer 2; and 4.) Battle Stations.

Mode Mixer 1 is a display list editor for Graphics modes 0-11 within an easy to understand menu. The MM 1 menu offers 8 OPTIONS, 4 of which will be used most often. The Create Screen Option gives you step input questions until you've filled 192 scan lines with whatever modes fit. The program shows how many scan lines remain after each input.

The User's Screen shows how the screen will be arranged. If you don't like something, you must go back and start over again. The Save to disk Option translates what you have done into BASIC statements. The Get Printed Copy Option prints the BASIC statements (if you have a printer).

You do have to remember to put the POKE statements into the right places in your program. The documentation shows you how to do this in a step-by-step example program. I was unable to get the Demo program to work on the disk I reviewed.

Mode Mixer 2 is a more advanced version of the editor. It requires more knowledge of scan lines and RAM, but it allows you to use more powerful features of the Atari. The extra Graphics modes include the Antic modes, multicolored texts and 4-color hi-res graphics (referred to as Graphics 12-16).

I'm sure you can see how this disk might save you a lot of time by not having to write the code for these routines, giving you more time to do things requiring more thought. M M 2 has no menu. It has a chart from which to make your choices, including the extra Antic modes. You will probably need a piece of scratch paper to figure out how many scan lines your new display will require. This is the hardest part of the program. A particularly nice feature allows you to automatically POKE memory past a 2k boundary when using Graphics 7 or 8 (for example). The program will also trace how much memory will be required for each graphics mode chosen.

Battle Stations lets you see what a good combination of graphic modes can do. This is a computer version of the game Battleship. The computer comes up with 2 grids where the players place ships in rows and columns. There is a Graphics mode 7 screen in the middle showing 5 ships for the players. Each time one hits a part of the other player's ship, there is an explosion, part of the ship disappears, and a red dot appears on the other player's grid. A miss makes a sound like something falling into water and a blue dot appears. Once all of one player's ships are destroyed, a ship sails across the screen with the words "YOU WIN".

I think just for the game alone the price is right. But you don't buy it for a game. This is a utility which will save you time and money. I believe this program will pay for itself each time you use it. Xlent Software did a good job of documentation, but the packaging leaves something to be desired. It is a well put together package for anyone wanting to put more graphics modes on the screen.

— Randy Cline

## GIVE YOUR OLD ATARI A SHOT . . .

(in the arm)

It was very lonely back in 1980 when I purchased my Atari 800. There were some computer buddies out there when I purchased my disk drive in 1981. I've been very happy with the unit but times change. My first indication of trouble with my system was this last Christmas when I received Loderunner by Broderbund as a gift. When I was finally able to load it after many tries, it would do everything but play. As soon as I began play it locked up.

I took the disk over to a friend who has a newer system and he booted it right up and began to play with no problem. So What's wrong with mine??? My computer still had the CTIA chip and revision "a" operating system (the OS board itself said rev "b" and my 810 didn't contain a data separator board).

I fixed my computer problems for about \$70 by purchasing 3 boards from : American TV Sales & Service, 1988 Washington Ave., San Leandro, CA 94577, (415) 352-3787. I purchased a REV "B" OS board (had REV "E" on it) a CPU board with a GTIA chip and a data separator board for my 810.

**WHAT A DIFFERENCE!!!** If you still have the old boards in your 800, I strongly recommend you purchase these new boards.

To determine if you have the old OS board type "PRINT PEEK(58383)". If the result is 56 then you have the old revision, 0 and you have the new one.

If you still have the box your 810 came in you can check it and see if it had a "DS" sticker on it. If so, then you have the data separator board. If you don't have the box you must check your 810 (the boards started being installed in early 1982). To check it you must remove the top of the 810 by removing the round tape covers from the 4 indentations in the top of the case and unscrewing the screws which lie underneath. When you remove the top you can see the 810 side board sitting vertically along the left side of the drive. There is a metal shield covering the middle of the side board. You must determine if there is a second board "piggybacked" on to the side board inside the shield. This may require you to remove the disk drive from the bottom of the case. The board, if there, is the data separator board and you don't need one. If there is not a second board inside the shield then by all means get one. It dramatically improves the drive's performance. If you do purchase the data separator board then be sure you bend the soldering connections on the underside of the board flat or they may hit the top of some IC's on the 810 side board which may block you from plugging in the DS board completely. If, after installing the board, you note that the drive only runs for about 1 sec (instead of about 3 now) when you turn it on then you don't have the DS board plugged into the 810 board completely. **DON'T** try running a disk in the drive if that is the case because you'll damage the disk.

The parts come with excellent installation instructions. Again, if you still have the old board do yourself a big favor and purchase these boards!!!

— Steve Berg

## FUN WITH ART

(reprinted from the Fresno Atari Computer Sector newsletter, April '84)

**FUN WITH ART** (Epyx Software, \$40, cartridge) is an excellent program with a great many capabilities for drawing. Not only is it powerful, but it's easy to use and on cartridge. After reading about this program and seeing some drawings created with it in ANTIC Magazine, I was eager to try it out for myself.

**FUN WITH ART** is a real joy to use, like all Epyx programs. Though the manual is only a few pages long and vague in its coverage of the features, the program is easily learned once the user works with it for a short time. Two thirds of the screen fills with a menu of icons (picture symbols). Below the menu area is a command window showing the commands available to the user. After studying the screen for a few minutes and reading the small instruction book, I am ready to go. I plug in the joystick and move the cursor around to the different icons. As I move the window indicated the choice under the cursor.

Well, let's see. I want to draw. So I pick the draw symbol on the far left. Now I'm ready to choose a color from the four colors shown at the bottom of the menu area. I can also choose to draw a color over or under another color. There is also a function to alter a color on the screen by using a display list interrupt. This is easily done by putting the cursor on the screen where the color change should start, then hitting the START key to move from the picture screen to the menu

screen. Next hitting the OPTION or SELECT key will change color and shade. To return to the picture screen, hit the START key again. The color should be changed from the cursor position down. This feature allows the user to put more colors on the screen than most people could ever dream imaginable. Just think, four colors on one line with the capabilities of putting two color changes at least one horizontal line apart. There are 24 design modes available with 128 colors from which to choose. Whew!! It takes a little time to take advantage of all this power in a drawing program.

Once you get used to these features you'll find yourself creating some very sophisticated drawings which you can load into your own BASIC programs using the listing I provide. This listing corrects the one provided with the Fun With Art program. Just run the program and type in the name of the picture file you wish to load. My listing also supports cassette picture storage, as well as disk storage. There is even a block storage feature which makes it easy to create screens full of graphics which can later be used to create future drawings.

All in all **FUN WITH ART** is the most impressive graphics drawing program for the ATARI computer I've seen. The best thing about the program is its cartridge form, a real plus for all the cassette system owners. So, if you like to draw or need a graphics program for your programming needs, **FUN WITH ART** is something you should seriously consider.



## Atari Computer Enthusiasts

A.C.E. is an independent, non-profit and tax exempt computer club and user's group with no connection to the Atari Company, a division of Warner Communication Company. We are a group interested in educating our members in the use of the Atari Computer and in giving the latest News, Reviews and Rumors.

**All our articles, reviews and programs come from you, our members.**

Our membership is world-wide; membership fees include the A.C.E. Newsletter. Dues are \$12 a year for U.S., and \$22 a year Overseas Air-mail and include about 10 issues a year of the ACE Newsletter.

**Subscription Dep't:** 3662 Vine Maple Dr., Eugene, OR 97405.

President— Kirt Stockwell, 4325 Sean , Eugene, OR 97402  
503-689-5355

Vice Pres— Larry Gold, 1927 McLean Blvd., Eugene, Or 97405  
503-686-1490

Secretary— Bruce Ebling, 1501 River Loop #1, Eugene, OR 97404  
503-688-6872

Librarian— Ron and Aaron Ness, 374 Blackfoot, Eugene, Or 97404  
(503)689-7106.

Editors— Mike Dunn, 3662 Vine Maple Dr., Eugene, Or 97405  
503-344-6193

Jim Bumpas, 4405 Dillard Rd., Eugene, Or 97405  
503-484-9925

E.R.A.C.E. (Education SIG Editor)-Ali Erickson, 295 Foxtail Dr., Eugene,  
OR 97405 /503-687-1133

E.R.A.C.E. Corresponding Secretary-Robert Browning, 90 W. Myoak,  
Eugene, OR 97404 (503)689-1513.

Send 50c stamps or coin (\$1 overseas) to the Ness' for the new, up-dated ACE Library List —new in May 84 !

### Best of ACE books

Volume 1 contains bound issues of the ACE Newsletter from the first issue, Oct 81 to June of 1982

Volume 2 covers July 1982 to June 1983

Only \$12 each (\$2 extra for Airmail). Available only from:

George Suetsugu  
45-602 Apuapu St  
Kaneohe, HI 96744

### SortFinder 1.2

A composite index of Atari related articles from 5 popular computer periodicals from Apr '81 to June '83, including ACE. Only \$6 for ACE member from:

Jim Carr, Valley Soft  
2660 S.W. DeArmond  
Corvallis, OR 97333

## TYPESETTING FROM YOUR COMPUTER

ATARI OWNERS: If you have a modem, text editor, and communications program to send ASCII files, you should consider the improved readability and cost savings provided by **TYPESETTING** your program documentation, manuscript, newsletter, or other lengthy text instead of just reproducing it from line printer or daisy-wheel output. Computer typesetting by telephone offers you high quality, space-saving copy that creates the professional image you want! Hundreds of type styles to choose from with 8 styles and 12 sizes "on line." And it's easy to encode your copy with the few typesetting commands you need.

### COMPLETE CONFIDENTIALITY GUARANTEED

— Bonded for your protection —

**PUBLICATION DESIGN, EDITING, & PRODUCTION**

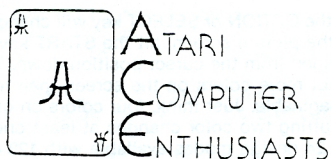
**Editing & Design Services**  
Inc.

**30 East 13th Avenue Eugene, Oregon 97401**  
**Phone 503/683-2657**

### Bulletin Board

(503) 343-4352

On line 24 hours a day, except for servicing and updating. Consists of a Tara equipped 48K Atari 400 with a TARA keyboard, 2 double-density double sided disk drives with an ATR 8000 interface, 2- 8" double density disk drives, an Epson MX80 printer, a Hayes SmartModem; running the ARMUDIC Bulletin Board software written by Frank L. Hubbard, 1206 N. Stafford St., Arlington, VA 22201. See the Nov '82 issue for complete details.



3662 Vine Maple Dr. Eugene OR 97405

# FIRST CLASS MAIL